

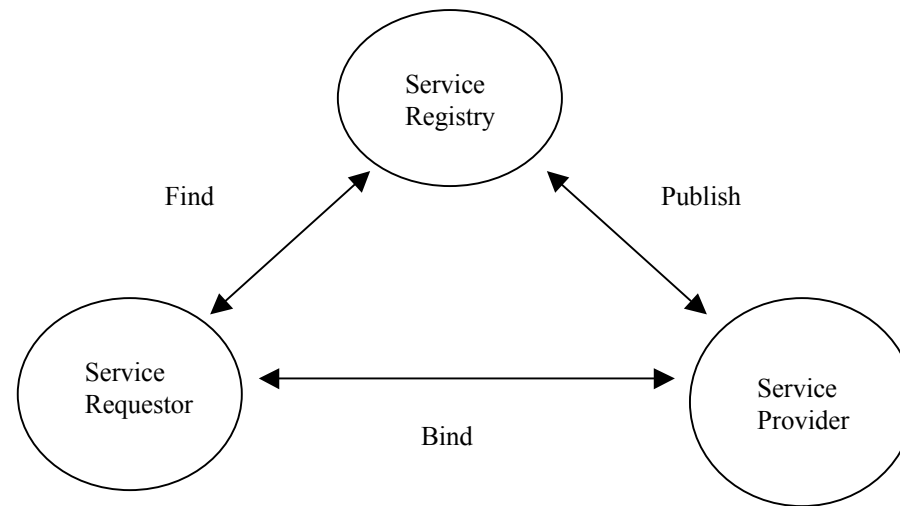
# Web Services

Seppo Heikkinen  
MITA seminar/TUT  
5.11.2003

# General overview

- Web services provide a standard means of interoperability between different software applications, running on a variety of platforms and/or frameworks.
- Machine readable interface definition
  - machine to machine interaction (program to program)
  - B2B
- Implementation independence
  - platform, language
  - wrapping of old legacy services
- Loosely coupled
  - distributed component orientation, reuse
  - dynamic service integration
- Service-oriented architecture (SOA)

# Conceptual model



- Service Provider hosts and publishes the service
- Service Registry is a searchable service description repository
  - can be seen to be a provider too
- Service Requestor finds suitable services and invokes them
  - can become a provider by registering itself

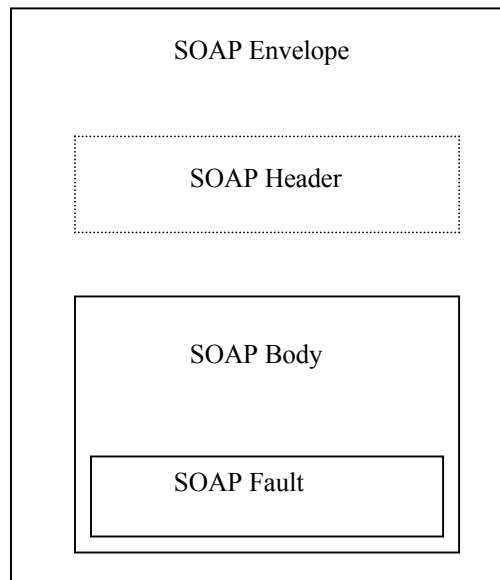
# Building blocks

- XML as common language (“lingua franca”)
- Communication using SOAP
- Service interfaces described with WSDL
- UDDI registry for discovery
  
- Technologies independent of each others
- Web Services Interoperability Organisation (WS-I) for enhancing interoperability and define the use of applicable technologies (e.g. Basic Profile which contains the above)

# SOAP

- Simple Object Access Protocol
  - pass structured and typed information between sender and receiver
  - XML based
    - poorer performance compared to binary protocols (ASN.1 binding might help)
- For accessing registries and invoking services (in WS context)
- Separate binding for transport protocol
  - could be HTTP, SMTP,...
- Fundamentally one-way messaging
  - typically request/response
  - can traverse several intermediaries
  - no multicast
- Remote procedure vs document oriented
  - end points define semantics

# SOAP message



SOAP message structure

```
POST /StockQuote HTTP/1.1
Host: www.stockquotesever.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "http://myservice.com/GetStock"

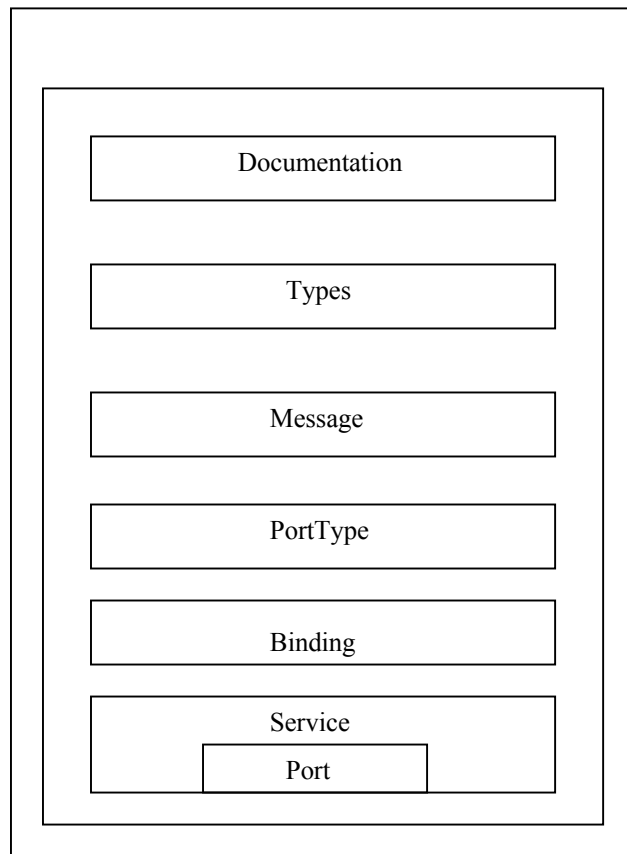
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Header>
    <t:Transaction
      xmlns:t="http://example1.com/"
      SOAP-ENV:mustUnderstand="1">
      5
    </t:Transaction>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <m:GetLastTradePrice xmlns:m="http://example2.com/">
      <symbol>DEF</symbol>
    </m:GetLastTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP 1.1 message example with  
HTTP transport (binding)

# WSDL

- Web Services Description Language
- “How and where to access the service”
- Describes the service interface definition
  - abstract messages (data types, formats)
    - usually XML Schema used
  - abstract access port for message operations
    - input/output
  - transport used (binding)
    - e.g. SOAP over HTTP
  - actual access points, i.e. network address of the service
    - implementation specific part
- Abstract vs concrete WSDL

# WSDL document structure



- Human readable description
- Data types
- Logical contents of messages
- Messages bound to abstract port
- Binding to transport
- Binding service port(s) to address

# WSDL example

```
<?xml version="1.0"?>
<definitions name="StockQuote"
  targetNamespace="http://example.com/stockquote.wsdl"
  xmlns:tns="http://example.com/stockquote.wsdl"
  xmlns:xsd1="http://example.com/stockquote.xsd"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <types>
    <schema targetNamespace="http://example.com/stockquote.xsd"
      xmlns="http://www.w3.org/2000/10/XMLSchema">
      <element name="TradePriceRequest">
        <complexType>
          <all>
            <element name="tickerSymbol" type="string"/>
          </all>
        </complexType>
      </element>
      <element name="TradePrice">
        <complexType>
          <all>
            <element name="price" type="float"/>
          </all>
        </complexType>
      </element>
    </schema>
  </types>

  <message name="GetLastTradePriceInput">
    <part name="body" element="xsd1:TradePriceRequest"/>
  </message>

  <message name="GetLastTradePriceOutput">
    <part name="body" element="xsd1:TradePrice"/>
  </message>

  <portType name="StockQuotePortType">
    <operation name="GetLastTradePrice">
      <input message="tns:GetLastTradePriceInput"/>
      <output message="tns:GetLastTradePriceOutput"/>
    </operation>
  </portType>
```

```
<binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="GetLastTradePrice">
    <soap:operation soapAction="http://example.com/GetLastTradePrice"/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>

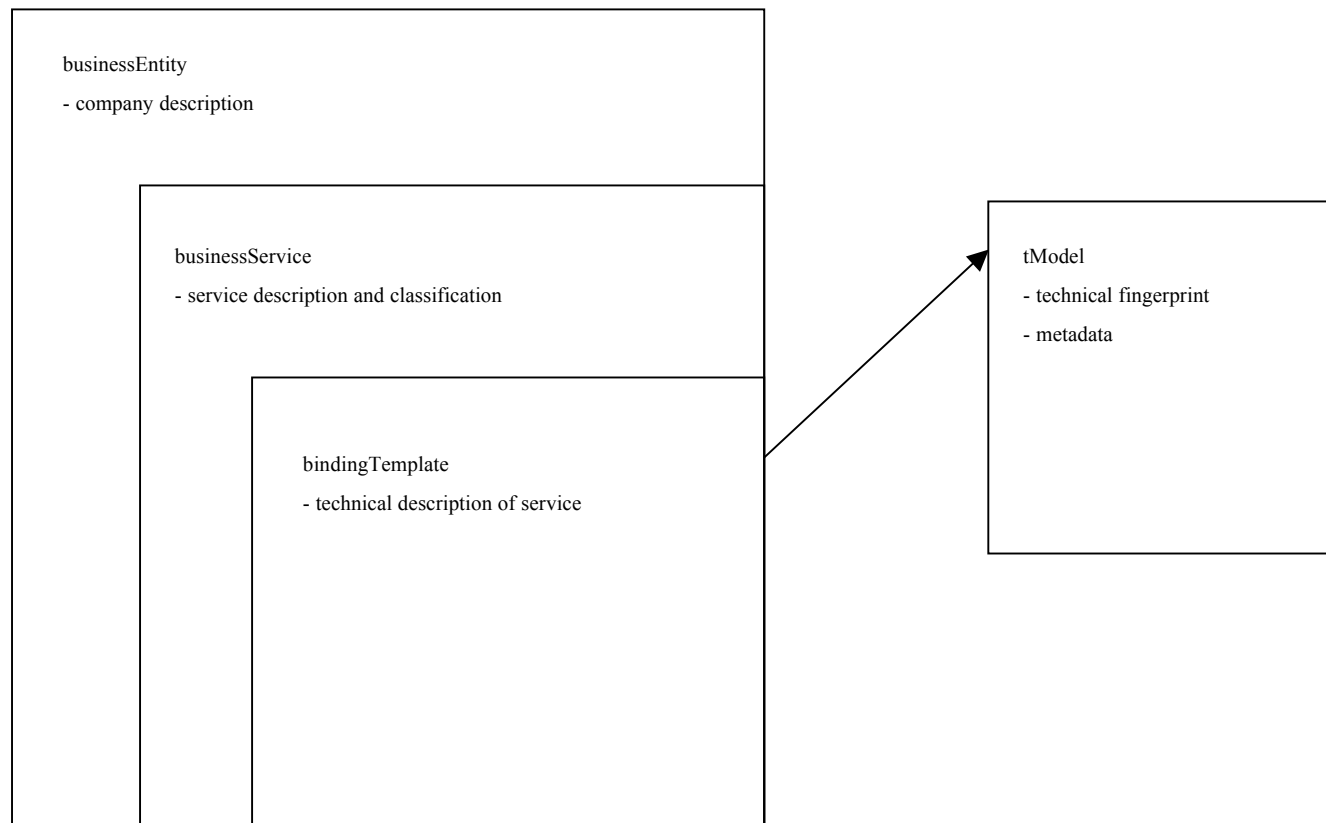
<service name="StockQuoteService">
  <port name="StockQuotePort" binding="tns:StockQuoteSoapBinding">
    <soap:address location="http://example.com/stockquote"/>
  </port>
</service>

</definitions>
```

# UDDI

- Universal Description, Discovery and Integration
- For publishing and searching businesses and services
  - make entities find each others
  - can be local or “global” registry
- White/yellow/green pages analogy
  - company contact information and description
  - classification with standardised taxonomies
  - technical information
    - service interface descriptions (e.g. locations where to find WSDL)
- Accessed using SOAP API

# UDDI structure



# UDDI example

```
<businessDetail generic="2.0" operator="www.ibm.com/services/uddi" truncated="false">
  <businessEntity businessKey="413E39E0-0807-11D8-B704-000629DC0A53"
operator="www.ibm.com/services/uddi">
  <discoveryURLs>
    <discoveryURL useType="businessEntity">
      http://uddi.ibm.com/ubr/uddiget?businessKey=413E39E0-0807-11D8-B704-000629DC0A53
    </discoveryURL>
  </discoveryURLs>
  <name xml:lang="en">Stock Company</name>
  <businessServices>
    <businessService serviceKey="B7E326A0-0807-11D8-B704-000629DC0A53"
businessKey="413E39E0-0807-11D8-B704-000629DC0A53">
      <name xml:lang="en">StockService</name>
      <description xml:lang="en">StockService_IBM testing</description>
      <bindingTemplates>
        <bindingTemplate bindingKey="B7F28FF0-0807-11D8-B704-000629DC0A53"
serviceKey="B7E326A0-0807-11D8-B704-000629DC0A53">
          <description xml:lang="en"/>
          <accessPoint URLType="http">
            http://www.example.com/StockService/StockService
          </accessPoint>
          <tModelInstanceDetails>
            <tModelInstanceInfo tModelKey="UUID:B55ACE10-0807-11D8-B704-
000629DC0A53"/>
          </tModelInstanceDetails>
          </bindingTemplate>
        </bindingTemplates>
        <categoryBag>
          <keyedReference tModelKey="UUID:C0B9FE13-179F-413D-8A5B-5004DB8E5BB2"
keyName="Portfolio Management" keyValue="52392"/>
        </categoryBag>
      </businessService>
    </businessServices>
  </businessEntity>
</businessDetail>
```

```
<tModelDetail generic="2.0" operator="www.ibm.com/services/uddi" truncated="false">
  <tModel tModelKey="UUID:B55ACE10-0807-11D8-B704-000629DC0A53"
operator="www.ibm.com/services/uddi">
  <name>StockService Specification</name>
  <description xml:lang="en">
    T-model for service interface definition
  </description>
  <overviewDoc>
    <overviewURL>
      http://www.example.com/StockService/StockService.wsdl
    </overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference tModelKey="UUID:C1ACF26D-9672-4404-9D70-39B756E62AB4"
keyName="uddi-org:types" keyValue="wsdlSpec"/>
  </categoryBag>
  </tModel>
</tModelDetail>
```

# Open issues

- Transactions, workflow, orchestration, choreography ?
  - Business processes and service composition
  - BPEL4WS
- Security ?
  - TLS/SSL for channel security
  - XML Signature, XML Encryption for message security
  - WS-Security draft for securing SOAP (OASIS)
  - Liberty Alliance, SAML
  - WS-Federation, WS-Trust,... (MS, IBM)
- QoS, reliability ?
- Accounting ?
- Alternatives: RosettaNet, ebXML
  - more business oriented approach

# Conclusions

- SOAP, WSDL and UDDI can be seen as low level enablers for distributed service platform
  - Interoperability
  - Application level still needs semantics
- Business processes need to be taken into consideration
- Web Services need standardised security solutions
- Standards convergence needed
- Web Services have potential (IDC predicts \$21 billion US market by 2007) but currently might be more suited for extra- and intranets

# References and further information

- MITA book volume 1, p. 109-126
- <http://www.w3.org/2002/ws/>
- <http://www.oasis-open.org/>
- <http://www-106.ibm.com/developerworks/webservices/>